In [1]:
```
%matplotlib inline

from matplotlib import pyplot as plt
import numpy as np
import imp
from IPython.display import YouTubeVideo
from IPython.display import HTML
```

EECS 545: Machine Learning

Lecture 23: Neural Networks (Part 2)

- Instructor: **Junhyuk Oh**
- Date: April 11, 2016

Outline

- Motivation
- Basics of Neural Networks
  - Forward Propagation
  - Backward Propagation
- Deep Neural Networks
  - Convolutional Neural Networks
  - **Recurrent Neural Networks**
- Applications
  - Computer Vision
  - Natural Language Processing
  - Reinforcement Learning

Outline: Recurrent Neural Networks

- Introduction
- Standard RNN
  - Forward Propagation
  - Backward Propagation
- Long Short-Term Memory (LSTM)
- Example: Character-level language modeling

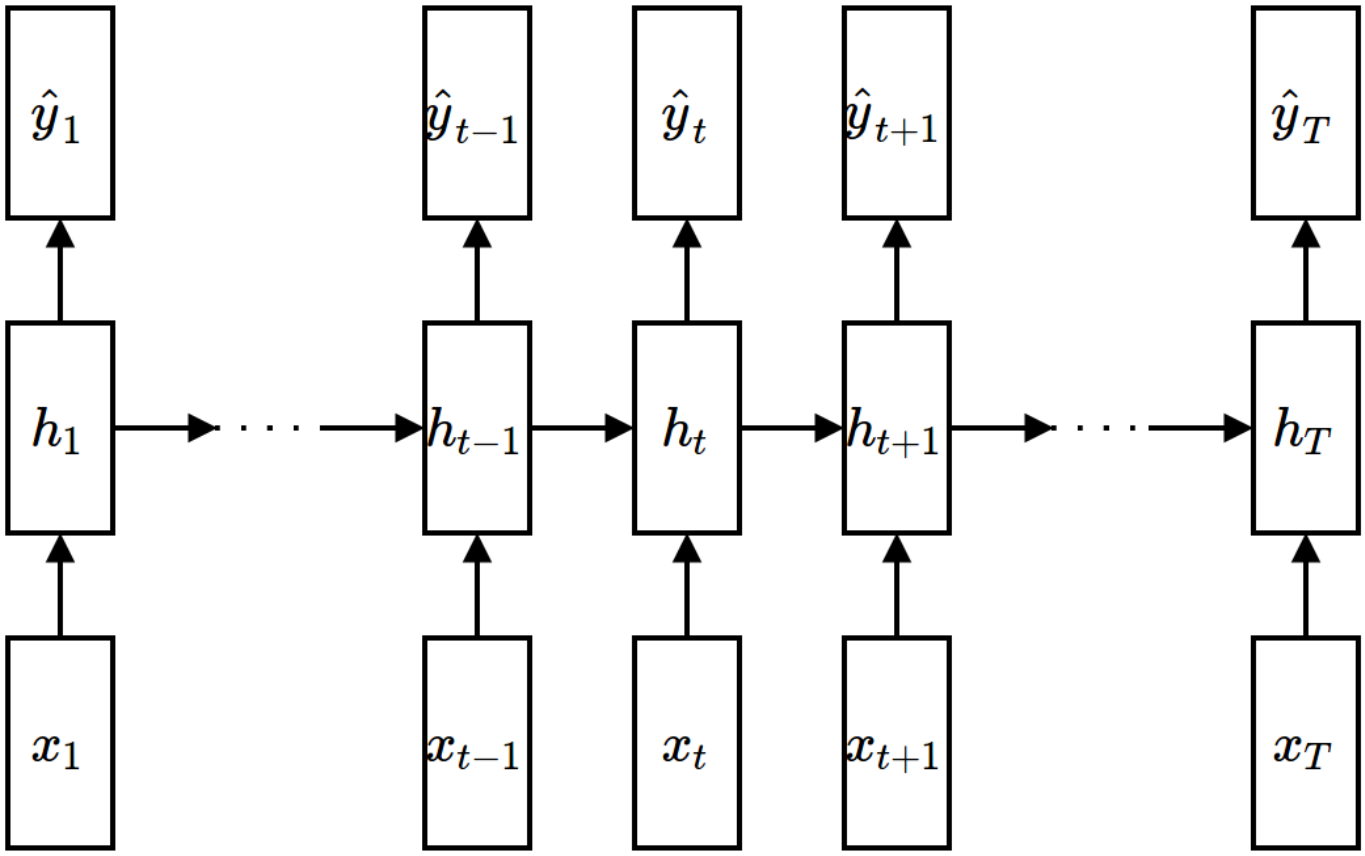
Recurrent Neural Networks (RNN)

- A special kind of neural network designed for modeling sequential data
  - Can take arbitrary number of inputs through time
  - Can produce arbitrary number of outputs through time
- Examples of sequential problems
  - Machine translation
  - Speech recognition
  - Image caption generaion

Outline: Recurrent Neural Networks

- Introduction
- **Standard RNN**
  - **Forward Propagation**
  - Backward Propagation
- Long Short-Term Memory (LSTM)
- Example: Character-level language modeling

Forward Propagation



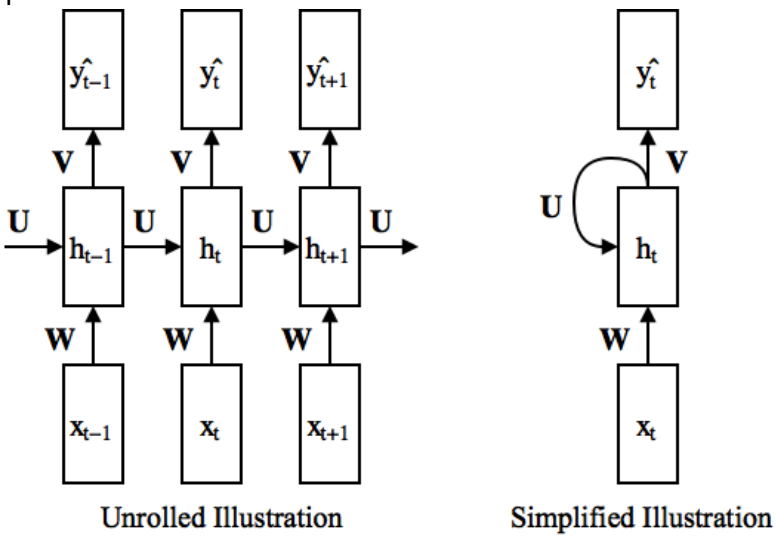
Forward Propagation

$$\mathbf{h}_t = f(\mathbf{W}\mathbf{x}_t + \mathbf{U}\mathbf{h}_{t-1} + \mathbf{b})$$

$$\hat{\mathbf{y}}_t = \mathbf{V}\mathbf{h}_t + \mathbf{b}'$$

$$\mathcal{L} = \sum_{t=1}^{T} \mathcal{L}_t \left(\mathbf{y}_t, \hat{\mathbf{y}}_t\right)$$

- $\mathbf{W}$: input weight, $\mathbf{U}$: recurrent weight, $\mathbf{V}$: output weight, $\mathbf{b}, \mathbf{b}'$: bias, f: non-linear activation (e.g., ReLU)
- Weights are shared across time: the number of parameters does not depend on the length of input/output sequence



Outline: Recurrent Neural Networks

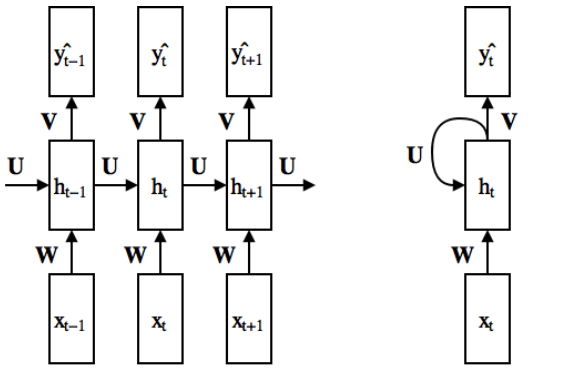
- Introduction
- Standard RNN
  - Forward Propagation
  - **Backward Propagation**
- Long Short-Term Memory (LSTM)
- Example: Character-level language modeling

Backpropagation Through Time (BPTT)

- Gradient w.r.t. hidden units (assuming that $\frac{\partial \mathcal{L}}{\partial \mathbf{h}_{t+1}}$ is given)

$$\frac{\partial \mathcal{L}}{\partial \mathbf{h}_t} = \sum_{\tau=t}^{T} \frac{\partial \mathcal{L}_\tau}{\partial \mathbf{h}_t} \; (\because \frac{\partial \mathcal{L}_k}{\partial \mathbf{h}_t} = 0 \text{ if } k < t)$$

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial \mathbf{h}_t} &= \frac{\partial \mathcal{L}_t}{\partial \mathbf{h}_t} + \frac{\partial \mathbf{h}_{t+1}}{\partial \mathbf{h}_t} \frac{\partial \sum_{\tau=t+1}^{T} \mathcal{L}_\tau}{\partial \mathbf{h}_{t+1}} \\ &= \underbrace{\frac{\partial \mathcal{L}_t}{\partial \hat{\mathbf{y}}_t} \frac{\partial \hat{\mathbf{y}}_t}{\partial \mathbf{h}_t}}_{\text{easy}\,\text{easy}} + \underbrace{\frac{\partial \mathbf{h}_{t+1}}{\partial \mathbf{h}_t} \frac{\partial \mathcal{L}}{\partial \mathbf{h}_{t+1}}}_{\text{easy}\quad\text{given}}\end{aligned}$$

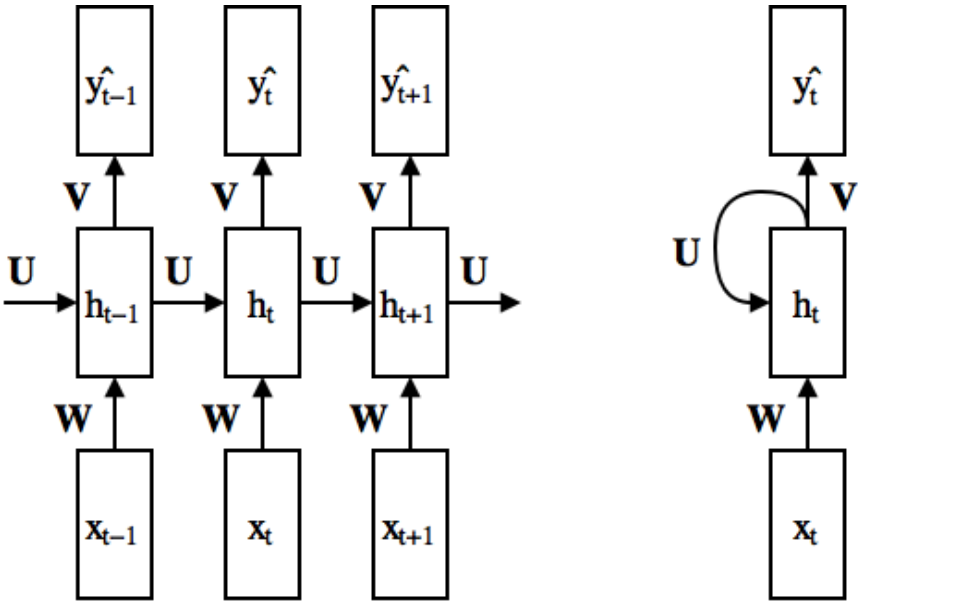


Unrolled Illustration Simplified Illustration

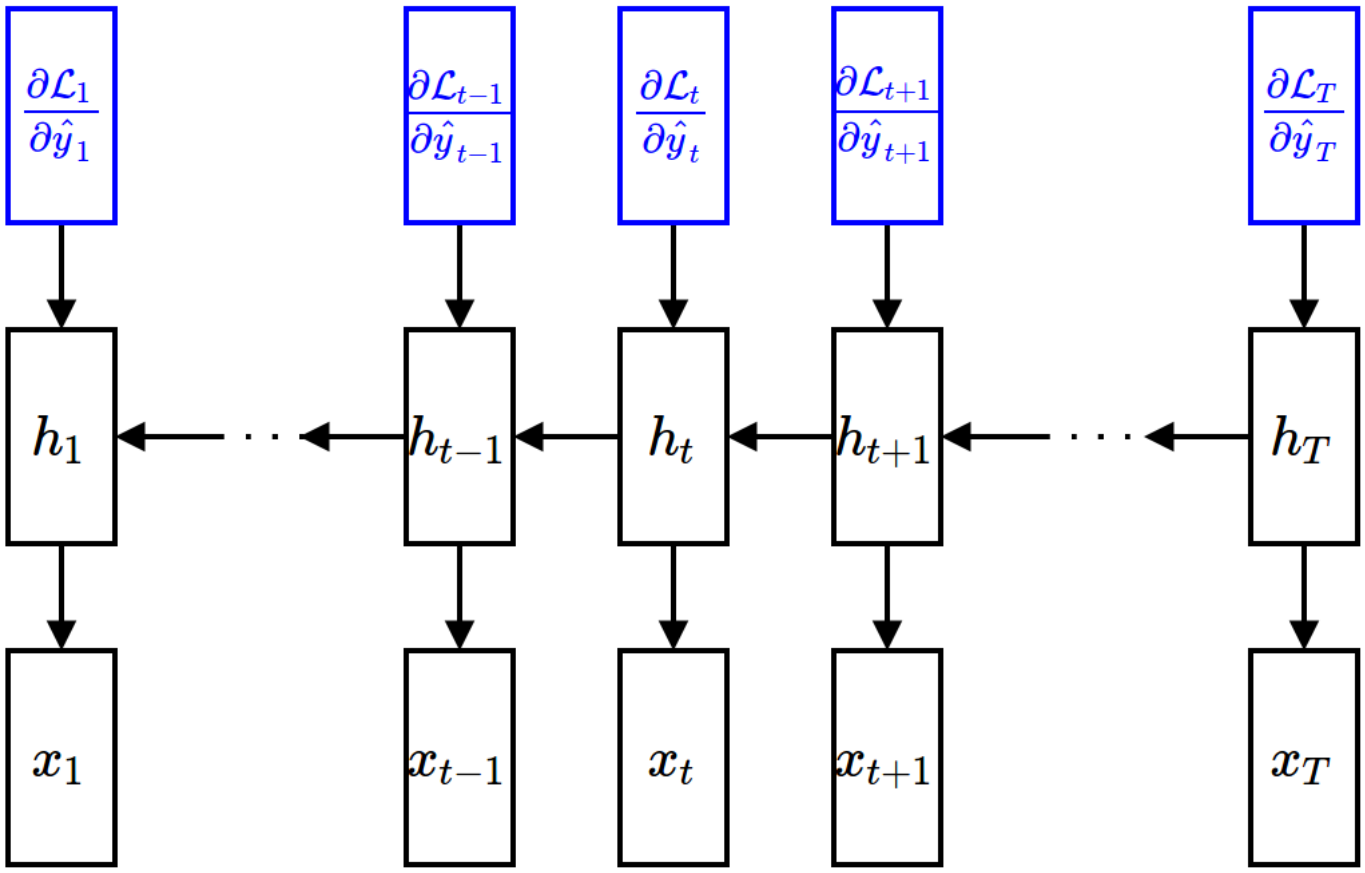
Backpropagation Through Time (BPTT)

- Gradient w.r.t. input units (given $\frac{\partial \mathcal{L}}{\partial \mathbf{h}_t}$)

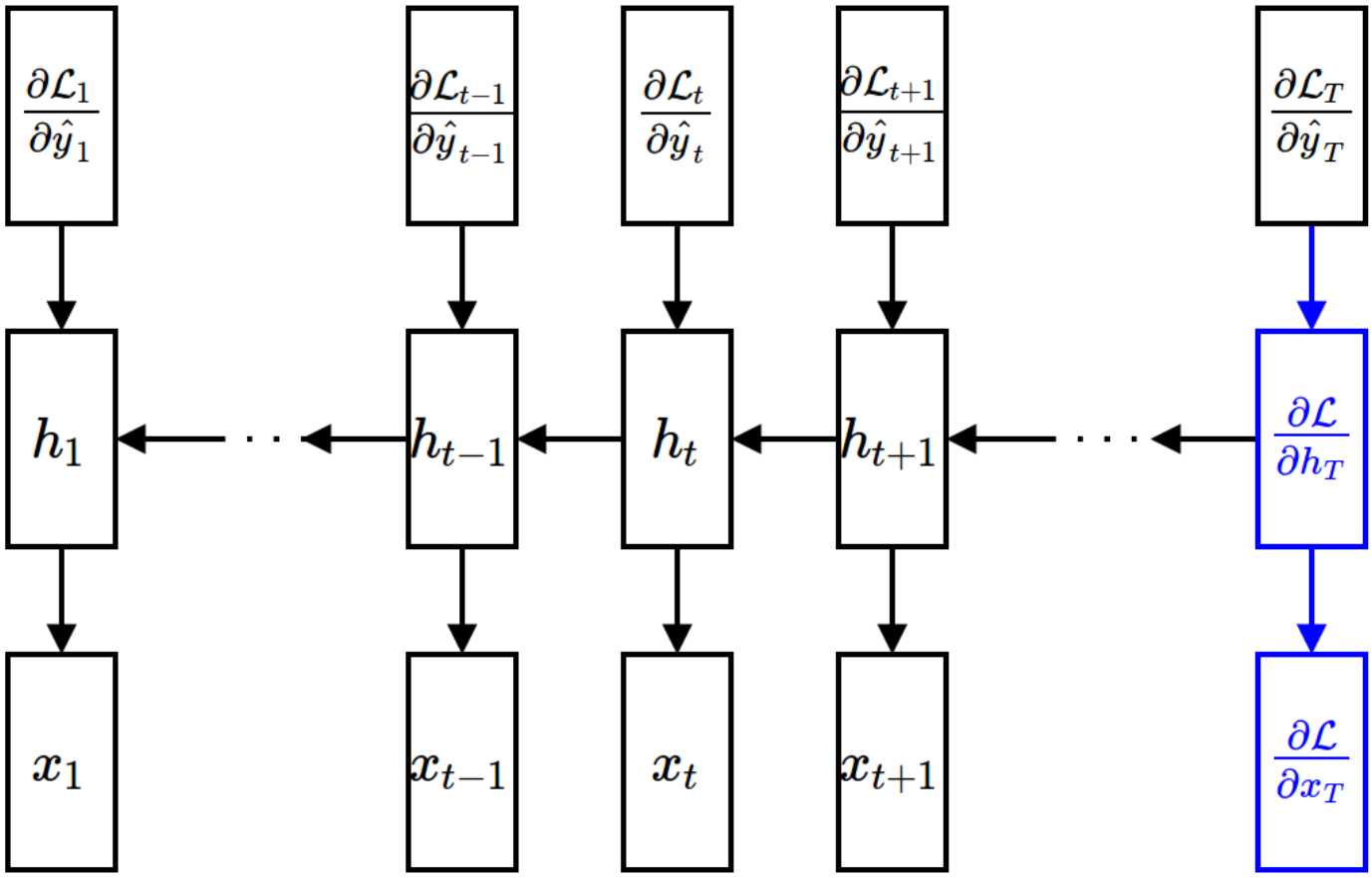
$$\frac{\partial \mathcal{L}}{\partial \mathbf{x}_t} = \frac{\partial \mathcal{L}}{\partial \mathbf{h}_t} \frac{\partial \mathbf{h}_t}{\partial \mathbf{x}_t}$$



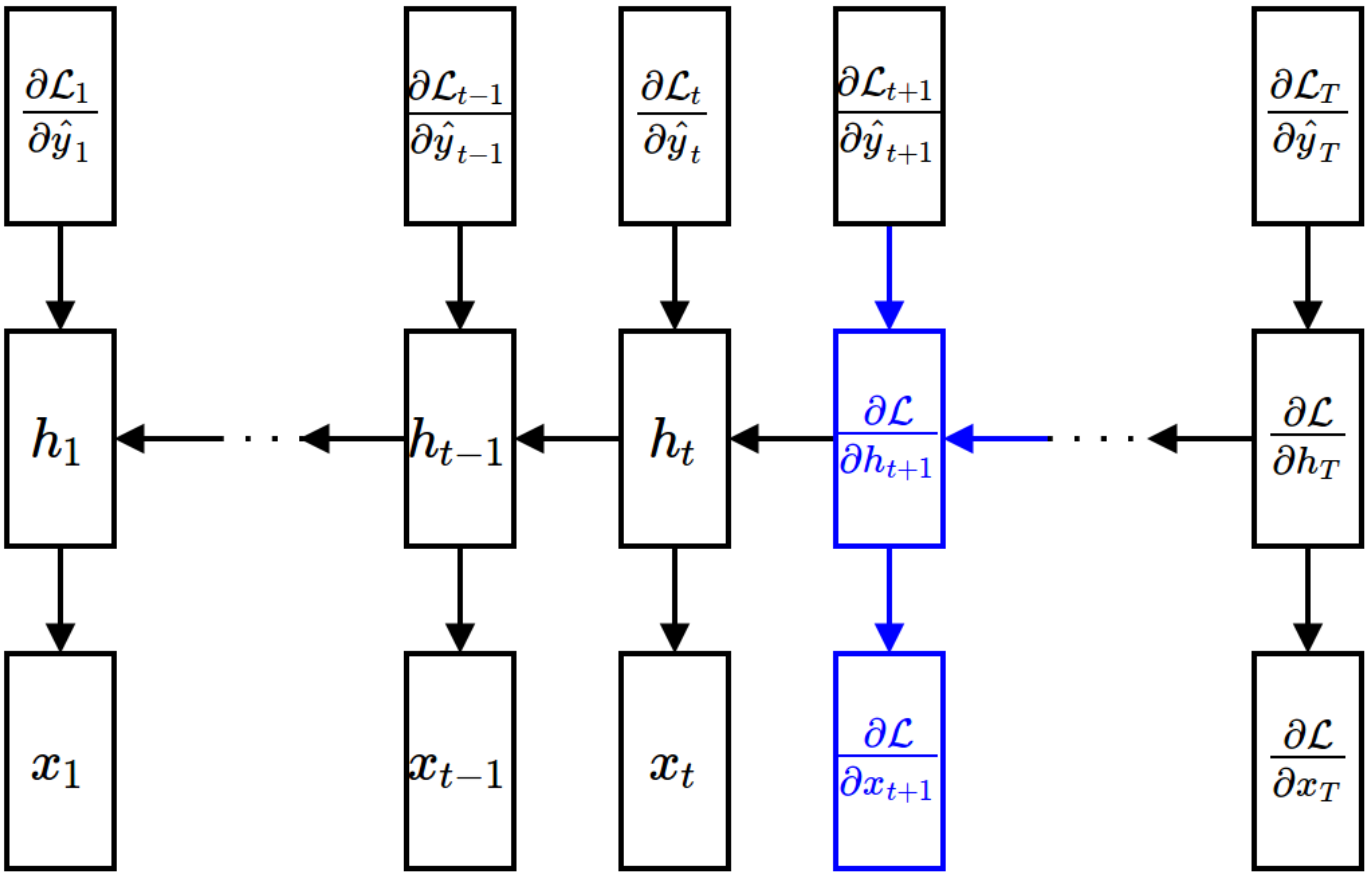
Backward Propagation



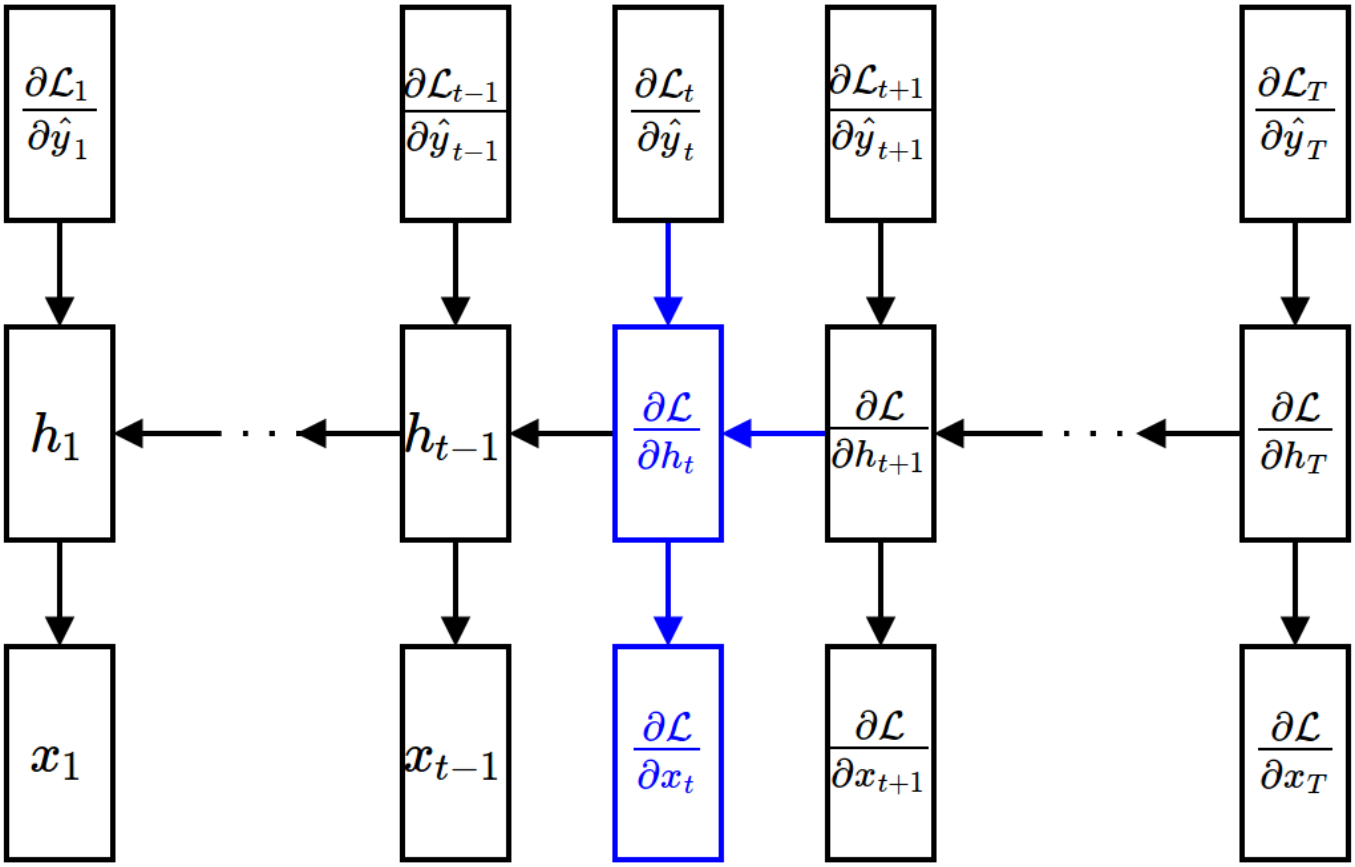
Backward Propagation



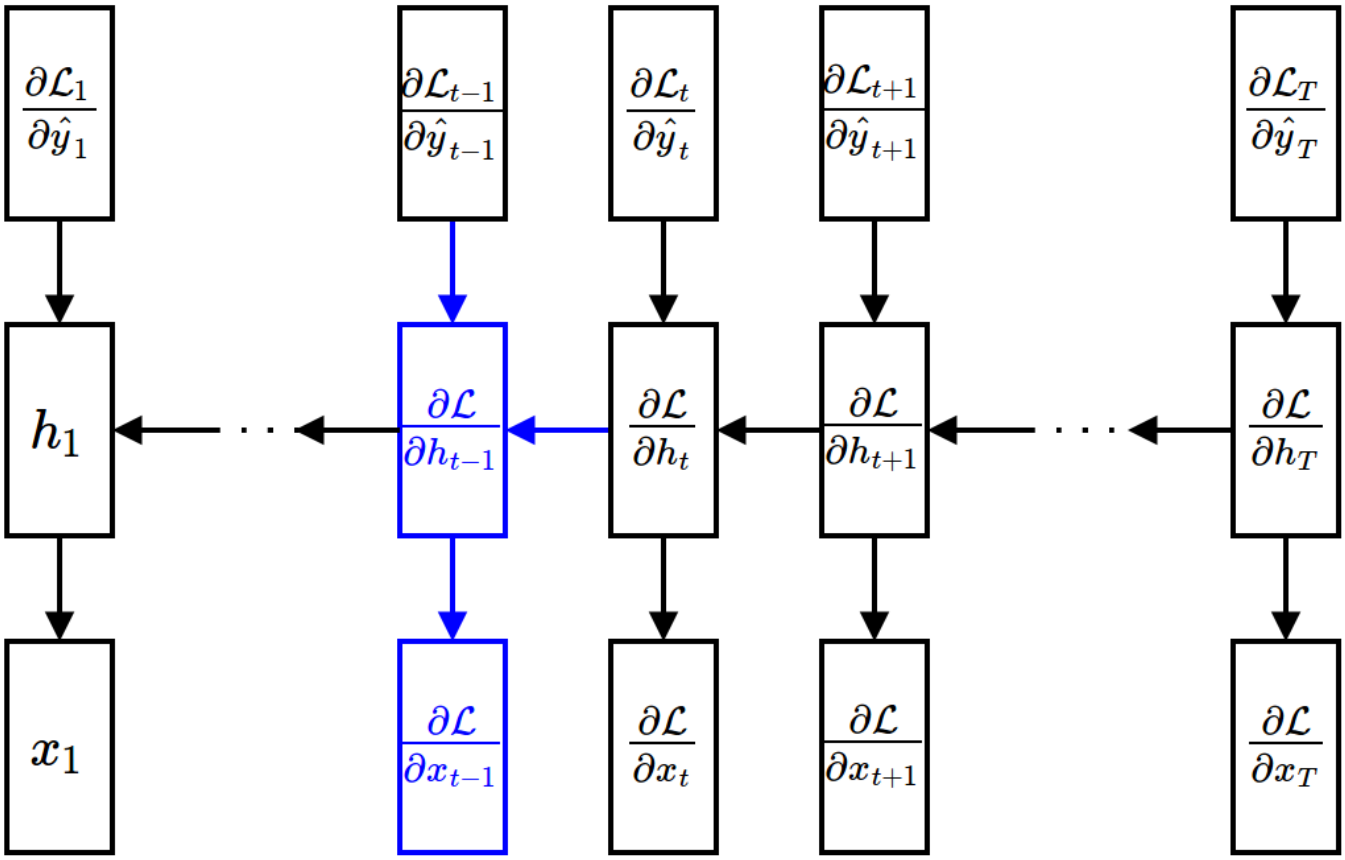
Backward Propagation



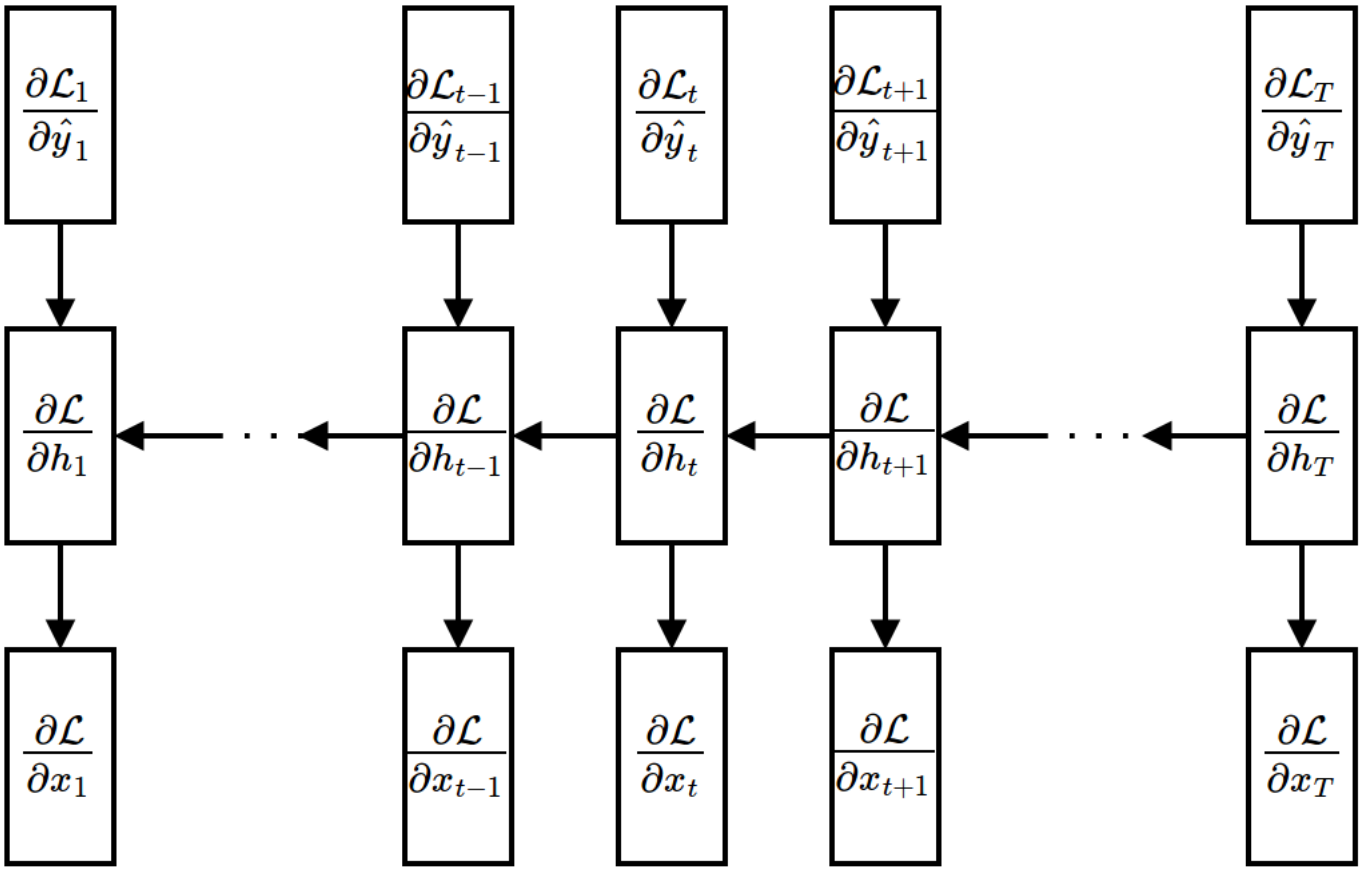
Backward Propagation



Backward Propagation



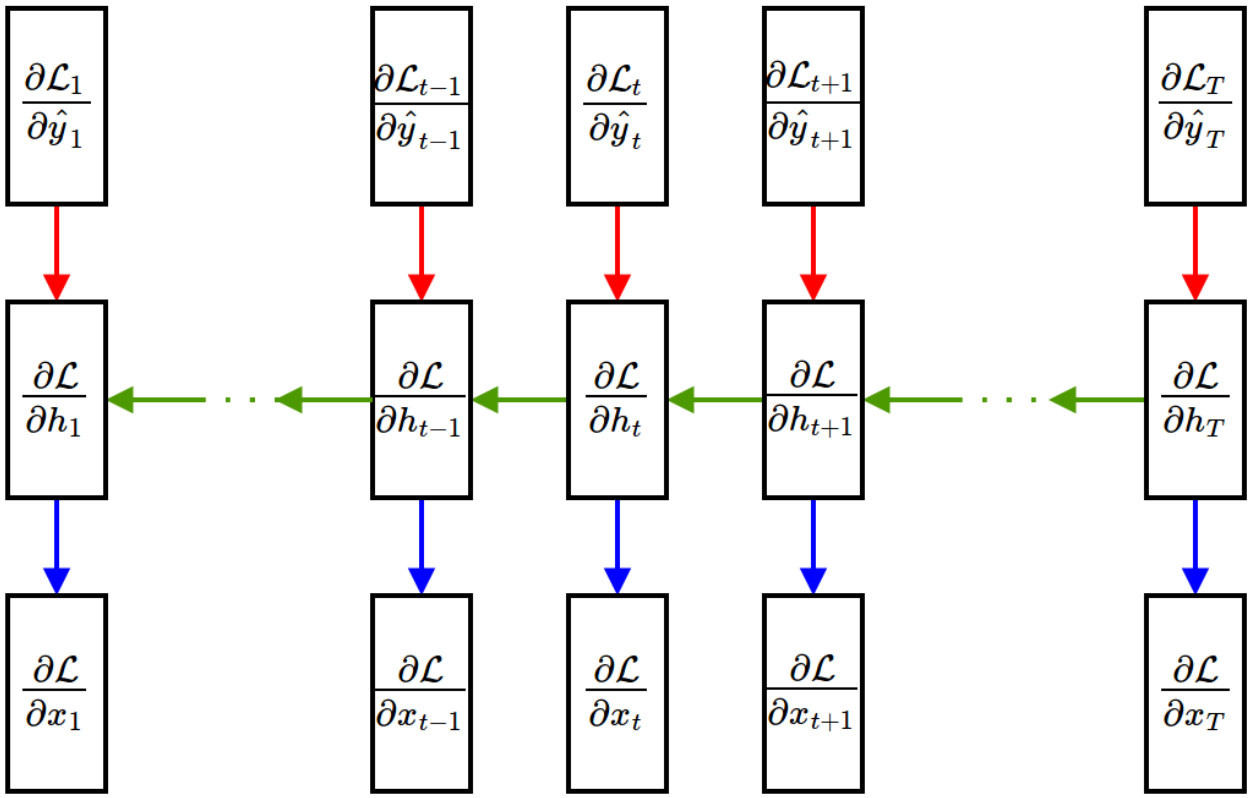
Backward Propagation



Backpropagation Through Time (BPTT)

- Gradient w.r.t. weights
  - Recall: The weights are shared through time. Gradients of shared weights should be accumulated!

$$\frac{\partial \mathcal{L}}{\partial \mathbf{V}} = \sum_{t=1}^{T} \frac{\partial \mathcal{L}}{\partial \hat{\mathbf{y}}_t} \frac{\partial \hat{\mathbf{y}}_t}{\partial \mathbf{V}} \quad \frac{\partial \mathcal{L}}{\partial \mathbf{W}} = \sum_{t=1}^{T} \frac{\partial \mathcal{L}}{\partial \mathbf{h}_t} \frac{\partial \mathbf{h}_t}{\partial \mathbf{W}} \quad \frac{\partial \mathcal{L}}{\partial \mathbf{U}} = \sum_{t=1}^{T-1} \frac{\partial \mathcal{L}}{\partial \mathbf{h}_{t+1}} \frac{\partial \mathbf{h}_{t+1}}{\partial \mathbf{U}}$$



Summary of Standard Recurrent Neural Network

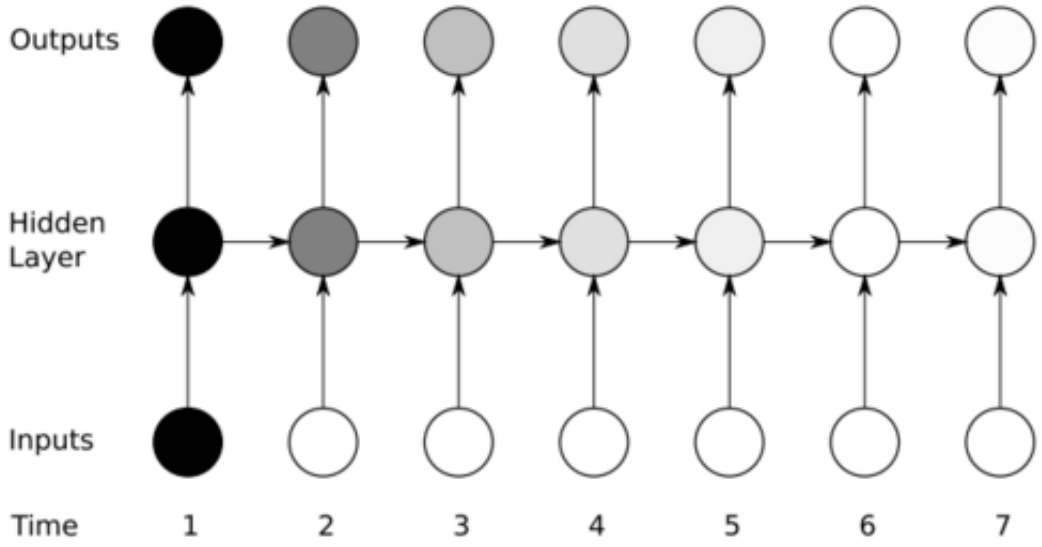
- RNN is actually not much different from a standard (feedforward) neural network except that:
  - Input/output are given through time.
  - Weights are extensively shared.
- RNN can be viewed as a very deep feedforward neural network with shared weights.

Outline: Recurrent Neural Networks

- Introduction
- Standard RNN
  - Forward Propagation
  - Backward Propagation
- **Long Short-Term Memory (LSTM)**
- Example: Character-level language modeling

Vanshing Gradient Problem

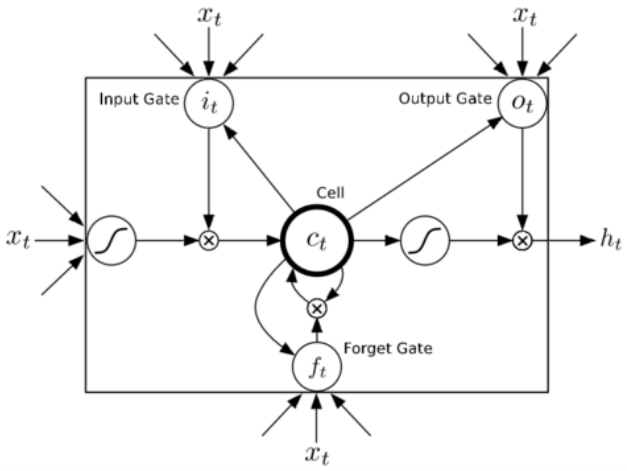
- RNN can model arbitrary sequences if properly trained.
- In practice, it is difficult to train it for long-term dependencies because of vanishing gradient.
- Intuition of vanishing gradient
  - A hidden unit activation is not well-preserved to the long-term future (forward propagation view)
  - Gradients are diffused through time (backward propagation view)



(Figure from Alex Graves)

Long Short-Term Memory (LSTM)

- A special type of RNN for handling vanishing gradient problem.
- i_t, o_t, f_t: **input gate**, **output gate**, and **forget gate**
- c_t: **memory cell** containing information about history of inputs
- h_t: output activation



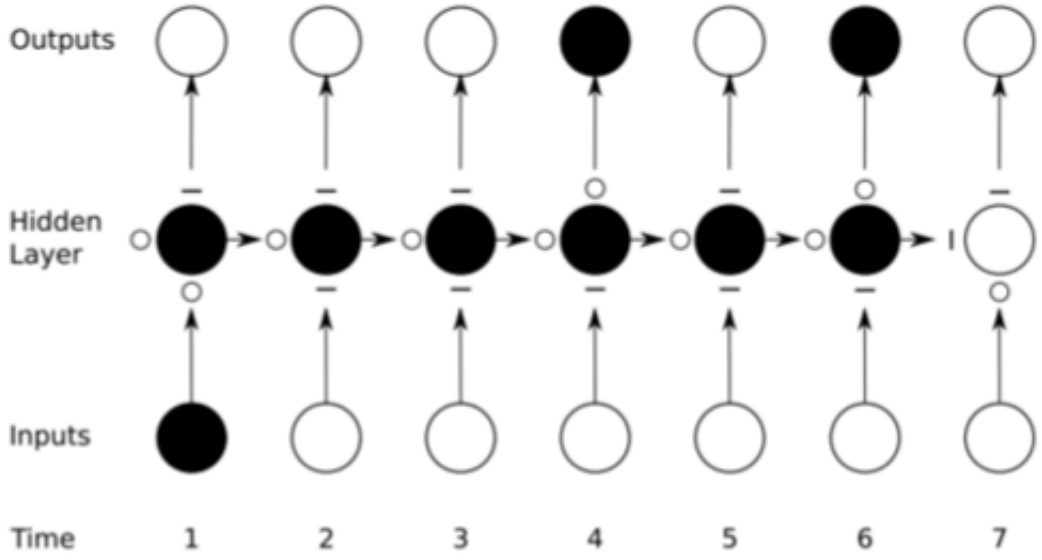
$$i_t = \sigma\left(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i\right)$$
$$f_t = \sigma\left(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f\right)$$
$$c_t = f_t c_{t-1} + i_t \tanh\left(W_{xc}x_t + W_{hc}h_{t-1} + b_c\right)$$
$$o_t = \sigma\left(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o\right)$$
$$h_t = o_t \tanh(c_t)$$

Input gate : scales input to cell (write)
Output gate : scales output from cell (read)
Forget gate : scales old cell value (reset)

(Figure from Alex Graves)

Long Short-Term Memory (LSTM)

- Gating mechanism
  - Input gate: whether to ignore a new input or not
  - Output gate: whether to produce an output or not (while preserving the memory cell)
  - Forget gate: whether to erase the memory cell or not
- Gating is controlled by LSTM's weights that are also learned from data.



(Figure from Alex Graves)

Outline: Recurrent Neural Networks

- Introduction
- Simple RNN
  - Forward Propagation
  - Backward Propagation
- Long Short-Term Memory (LSTM)
- **Example: Character-level language modeling**

Character-level language modeling

- Goal: build a character-level generative model
- A character ($\boldsymbol{x}_t$) is represented as a one-of-k vector (k: #characters).

$$P(x_1, x_2, \ldots, x_T) = \prod_{t=1}^{T} P(x_t | x_{t-1}, \ldots, x_1) \approx \prod_{t=1}^{T} P(x_t | x_{t-1}, \ldots, x_{t-n})$$

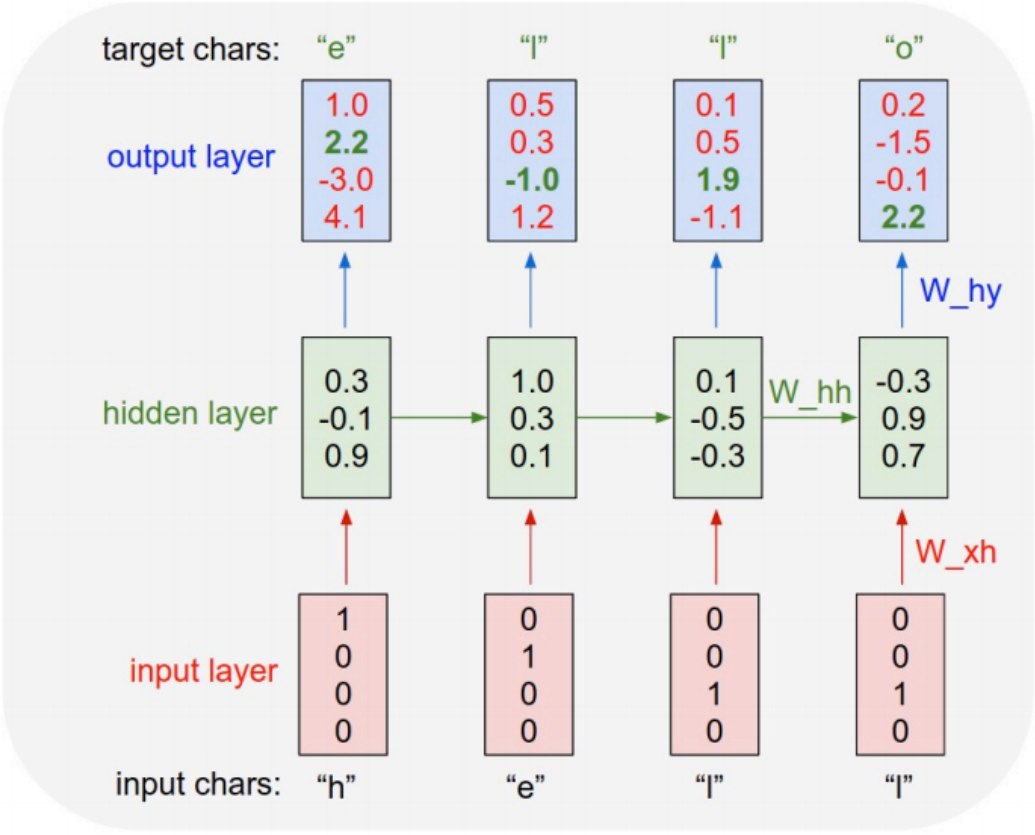
Character-level language modeling

- RNN is trained to predict next character given previous characters
- Maximizing the likelihood can be formulated as minimizing sum of cross entropy losses.

$$\mathcal{L}(\mathbf{x}) = -\sum_{t=1}^{T} \log P(x_t | x_{t-1}, \ldots, x_{t-n}; \theta)$$

Character-level language modeling

- After training, the network can "sample" characters from the multinomial distribution of characters at the output layer (softmax).



http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Shakespeare

PANDARUS:
Alas, I think he shall be come approached and the day
When little srain would be attain'd into being never fed,
And who is but a chain and subjects of his death,
I should not sleep.

Second Senator:
They are away this miseries, produced upon my soul,
Breaking and strongly should be buried, when I perish
The earth and thoughts of many states.

DUKE VINCENTIO:
Well, your wit is in the care of side and that.

(Figure from Richard Socher)

Wikipedia

Naturalism and decision for the majority of Arab countries' capitalide was grounded by the Irish language by [[John Clair]], [[An Imperial Japanese Revolt]], associated with Guangzham's sovereignty. His generals were the powerful ruler of the Portugal in the [[Protestant Immineners]], which could be said to be directly in Cantonese Communication, which followed a ceremony and set inspired prison, training. The emperor travelled back to [[Antioch, Perth, October 25|21]] to note, the Kingdom of Costa Rica, unsuccessful fashioned the [[Thrales]], [[Cynth's Dajoard]], known in western [[Scotland]], near Italy to the conquest of India with the conflict.

(Figure from Richard Socher)

Latex

Proof. Omitted. □

Lemma 0.1. *Let $\mathcal{C}$ be a set of the construction.*

Let $\mathcal{C}$ be a gerber covering. Let $\mathcal{F}$ be a quasi-coherent sheaves of $\mathcal{O}$-modules. We have to show that

$$\mathcal{O}_{\mathcal{O}_X} = \mathcal{O}_X(\mathcal{L})$$

.

Proof. This is an algebraic space with the composition of sheaves $\mathcal{F}$ on $X_{\acute{e}tale}$ we have

$$\mathcal{O}_X(\mathcal{F}) = \{morph_1 \times_{\mathcal{O}_X} (\mathcal{G}, \mathcal{F})\}$$

where $\mathcal{G}$ defines an isomorphism $\mathcal{F} \to \mathcal{F}$ of $\mathcal{O}$-modules. □

Lemma 0.2. *This is an integer $\mathcal{Z}$ is injective.*

Proof. See Spaces, Lemma **??**. □

Lemma 0.3. *Let S be a scheme. Let X be a scheme and X is an affine open covering. Let $\mathcal{U} \subset \mathcal{X}$ be a canonical and locally of finite type. Let X be a scheme. Let X be a scheme which is equal to the formal complex.*

The following to the construction of the lemma follows.

Let X be a scheme. Let X be a scheme covering. Let

$$b : X \to Y' \to Y \to Y \to Y' \times_X Y \to X.$$

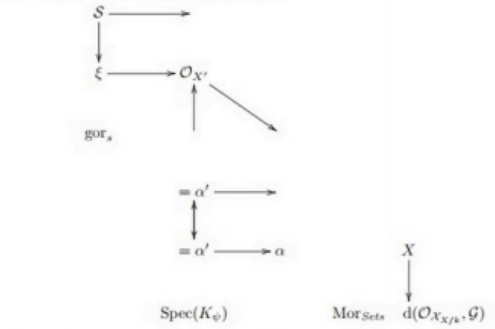
be a morphism of algebraic spaces over S and Y.

Proof. Let X be a nonzero scheme of X. Let X be an algebraic space. Let $\mathcal{F}$ be a quasi-coherent sheaf of $\mathcal{O}_X$-modules. The following are equivalent

(1) $\mathcal{F}$ is an algebraic space over S.
(2) If X is an affine open covering.

Consider a common structure on X and X the functor $\mathcal{O}_X(U)$ which is locally of finite type. □

This since $\mathcal{F} \in \mathcal{F}$ and $x \in \mathcal{G}$ the diagram



is a limit. Then $\mathcal{G}$ is a finite type and assume S is a flat and $\mathcal{F}$ and $\mathcal{G}$ is a finite type f_*. This is of finite type diagrams, and

- the composition of $\mathcal{G}$ is a regular sequence,
- $\mathcal{O}_{X'}$ is a sheaf of rings.

□

Proof. We have see that $X = \mathrm{Spec}(R)$ and $\mathcal{F}$ is a finite type representable by algebraic space. The property $\mathcal{F}$ is a finite morphism of algebraic stacks. Then the cohomology of X is an open neighbourhood of U. □

Proof. This is clear that $\mathcal{G}$ is a finite presentation, see Lemmas **??**.
A *reduced above* we conclude that U is an open covering of $\mathcal{C}$. The functor $\mathcal{F}$ is a "field

$$\mathcal{O}_{X,x} \longrightarrow \mathcal{F}_{\overline{x}} \quad \text{-1}(\mathcal{O}_{X_{\acute{e}tale}}) \longrightarrow \mathcal{O}_{X_\lambda}^{-1}\mathcal{O}_{X_\lambda}(\mathcal{O}_{X_\eta}^{\overline{v}})$$

is an isomorphism of covering of $\mathcal{O}_{X_i}$. If $\mathcal{F}$ is the unique element of $\mathcal{F}$ such that X is an isomorphism.
The property $\mathcal{F}$ is a disjoint union of Proposition **??** and we can filtered set of presentations of a scheme $\mathcal{O}_X$-algebra with $\mathcal{F}$ are opens of finite type over S.
If $\mathcal{F}$ is a scheme theoretic image points. □

If $\mathcal{F}$ is a finite direct sum $\mathcal{O}_{X_\lambda}$ is a closed immersion, see Lemma **??**. This is a sequence of $\mathcal{F}$ is a similar morphism.

(Figure from Richard Socher)

C++ Code

```
/*
 * Increment the size file of the new incorrect UI_FILTER group information
 * of the size generatively.
 */
static int indicate_policy(void)
{
  int error;
  if (fd == MARN_EPT) {
    /*
     * The kernel blank will coeld it to userspace.
     */
    if (ss->segment < mem_total)
      unblock_graph_and_set_blocked();
    else
      ret = 1;
    goto bail;
  }
  segaddr = in_SB(in.addr);
  selector = seg / 16;
  setup_works = true;
```

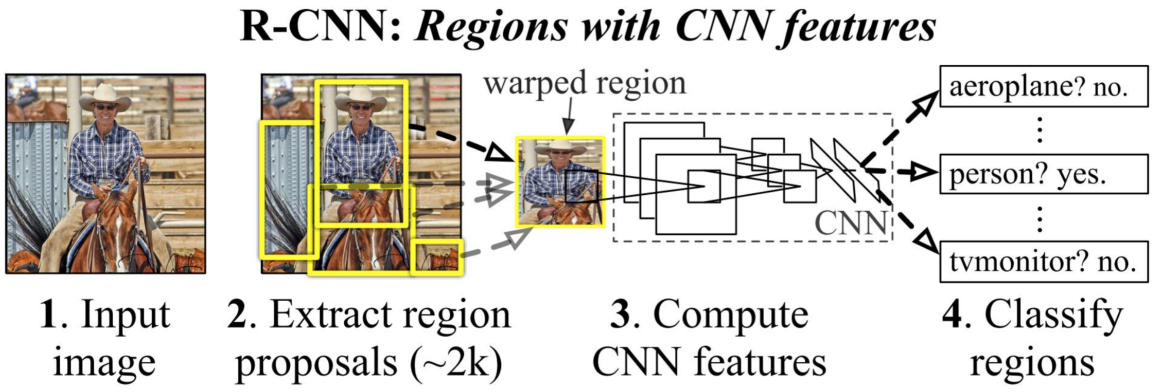
(Figure from Richard Socher)

Outline

- Motivation
- Basics of Neural Networks
  - Forward Propagation
  - Backward Propagation
- Deep Neural Networks
  - Convolutional Neural Networks
  - Recurrent Neural Networks
- Applications
  - **Computer Vision**
  - Natural Language Processing
  - Reinforcement Learning

Object Detection

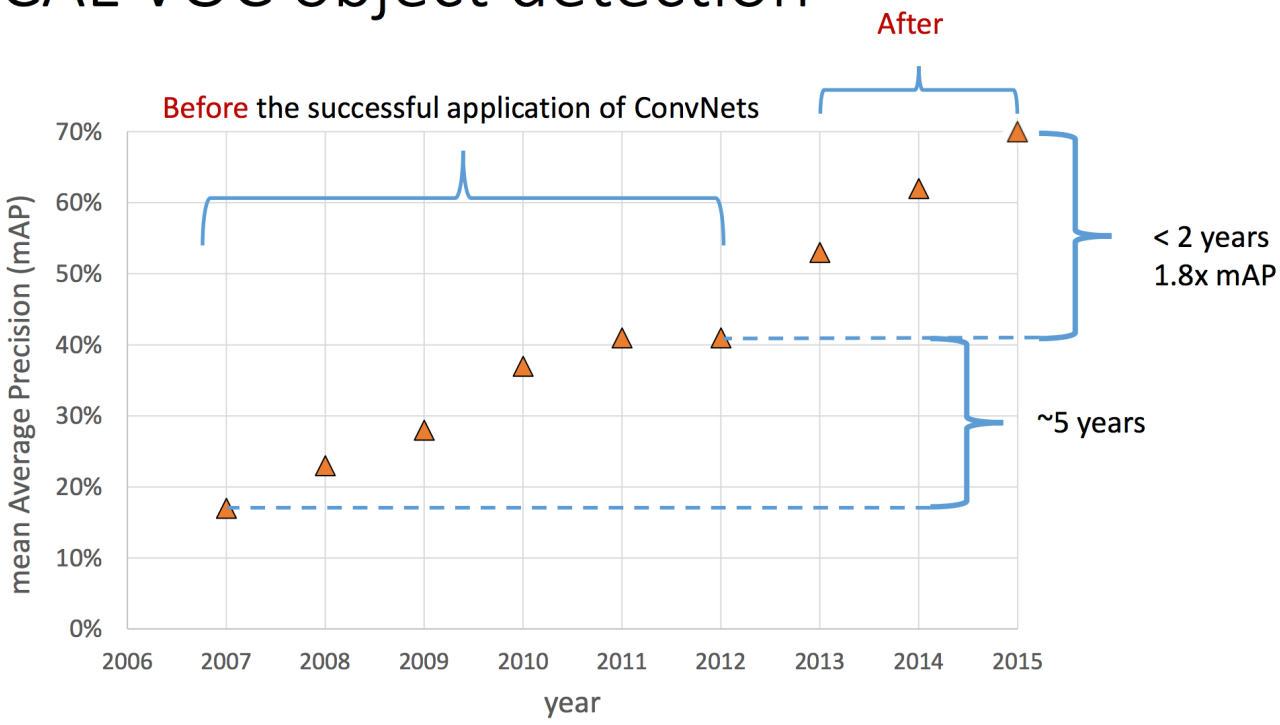
- Goal: find bounding boxes that contain objects in an image and predict their classes.
- CNN approaches have recently achieved state-of-the-art results on object detection task.
- Example: Regions with CNN
  - Use a low-level region proposal methods to generate many candidate bounding boxes
  - Use a CNN to do classification for each region



(Girshick et al, "Reigion-based Convolutional Networks for Accurate Object Detection and Semantic Segmentation", PAMI, 2015.)

Object Detection

PASCAL VOC object detection

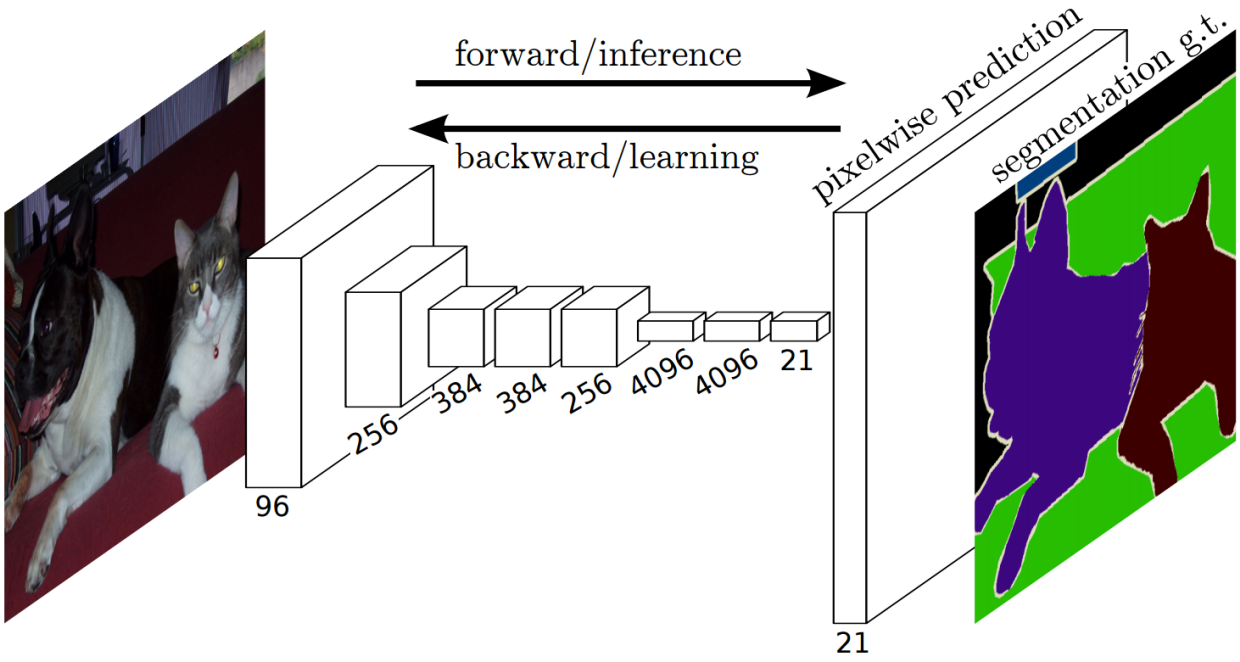


Precision: higher is better

(Figur from Ross Girshick)

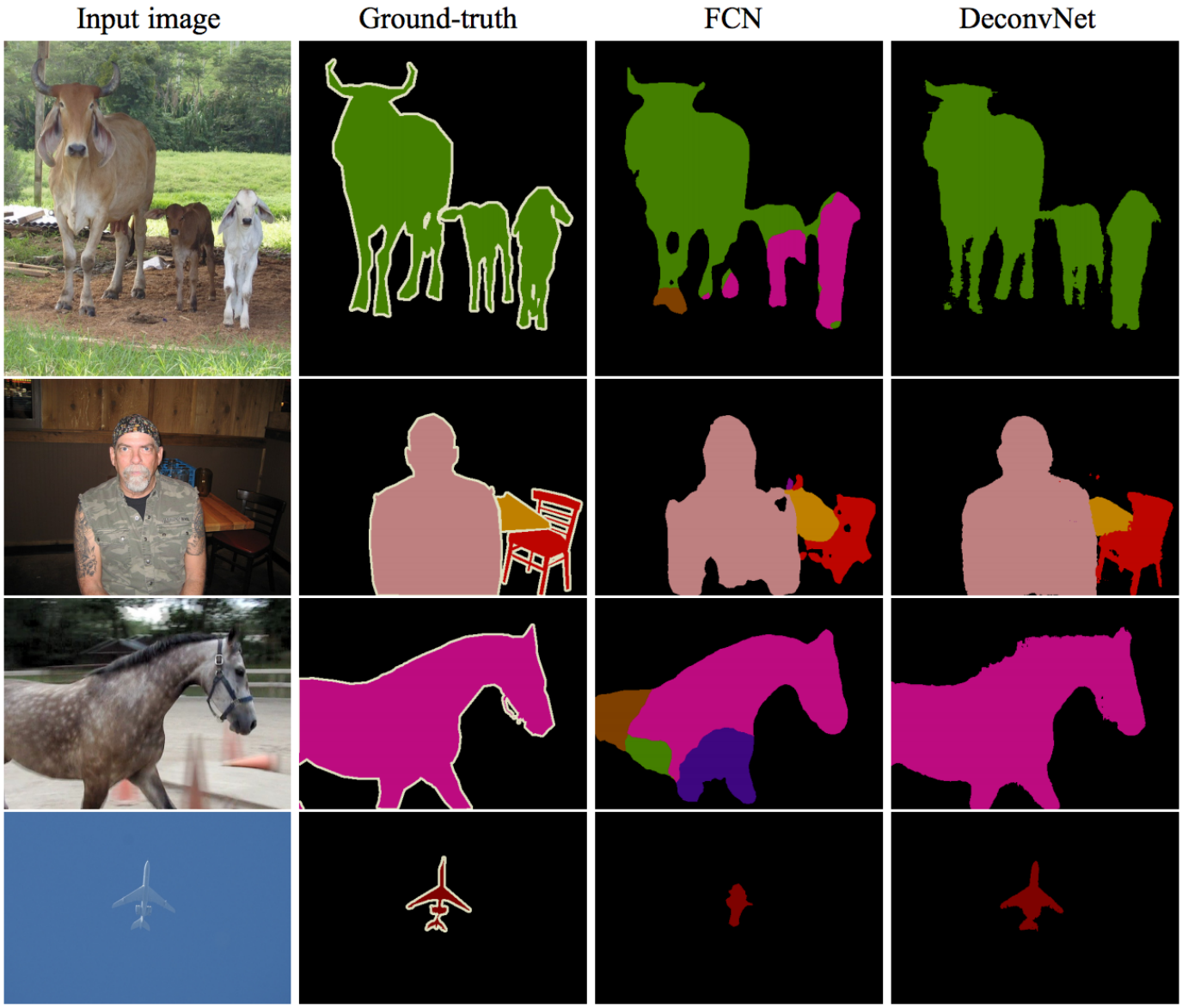
Object Segmentation

- Goal: Segment object regions and predict class labels for each region
- Can be formulated as pixel-wise classification
- Pre-trained on a large-scale classification dataset (ImageNet)



(Long et al, "Fully Convolutional Networks for Semantic Segmentation", CVPR, 2015.)

Object Segmentation



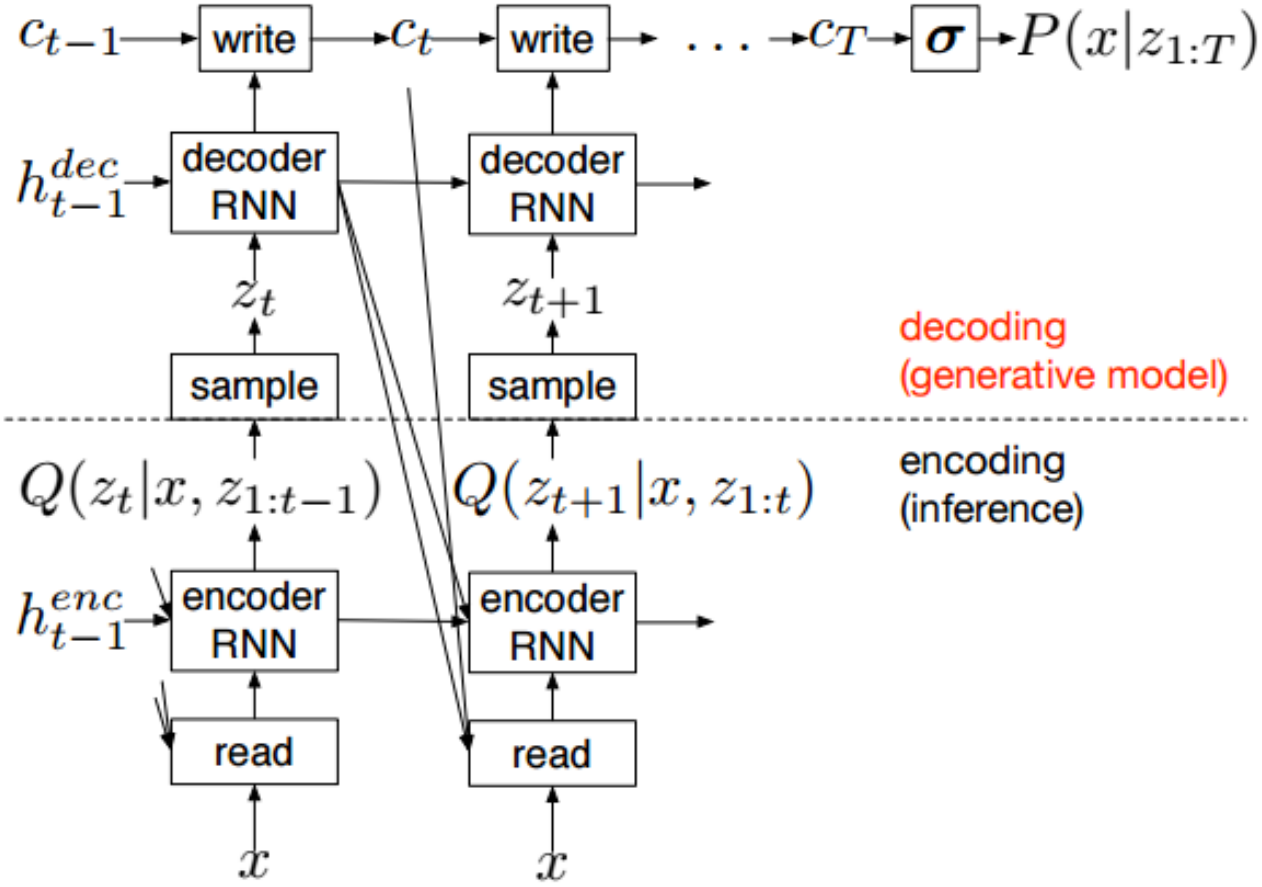
(Noh et al, "Learning Deconvolution Network for Semantic Segmentation", ICCV, 2015.)

Image Generation

- Goal: $\max_\theta \mathbb{E}_{\mathbf{x}} \left[\log P\left(\mathbf{x};\theta\right)\right]$
- Outcome: $\mathbf{x} \sim P\left(\mathbf{x};\theta\right)$

Image Generation: DRAW Network (Gregor et al.)

- Introduce a differentiable visual attention mechanism
- Learn to move an attention window during image generation process (without any supervision).



(Gregor et al, "DRAW: A Recurrent Neural Network For Image Generation", ICML, 2015.)

Image Generation: DRAW Network (Gregor et al.)

In [2]:
```
YouTubeVideo("Zt-7MI9eKEo", width=800, height=600)
```

Out[2]:

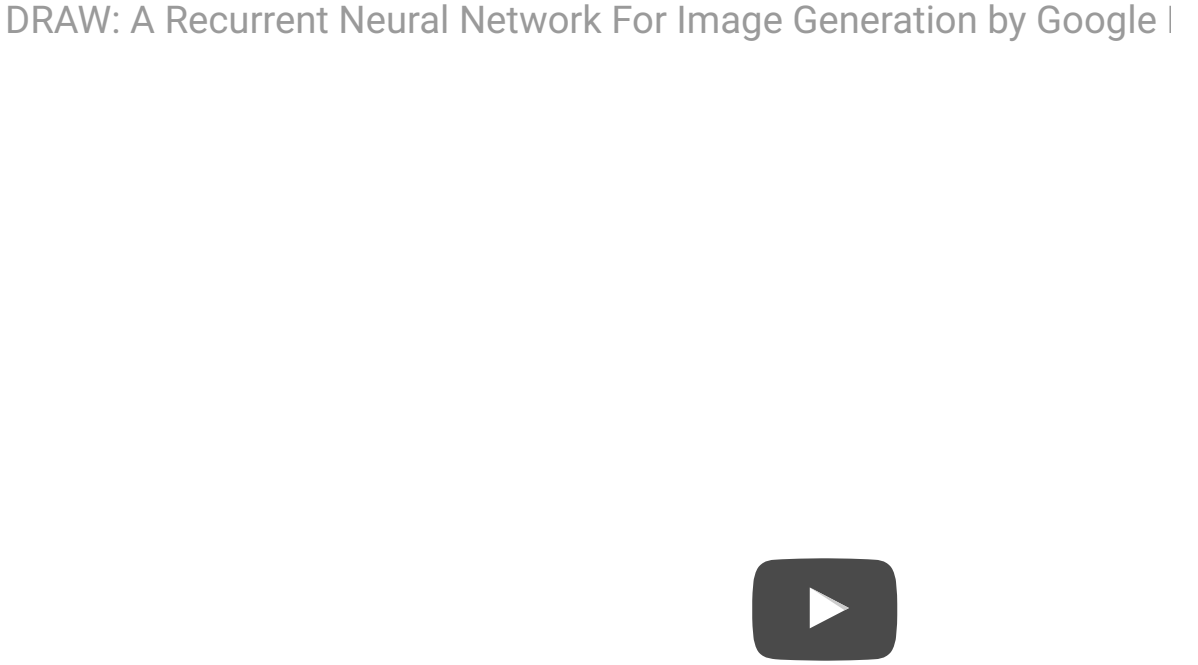


Image Generation: Adversarial Network (Goodfellow et al., Radford et al.)

- Model : Generator (G), Discriminator (D)
- Generator (G) : Learns to generate realistic images
- Discriminator (D) : Learns to classify whether a given image is real (from data) or not (from model)
- Objective : Fool each other!

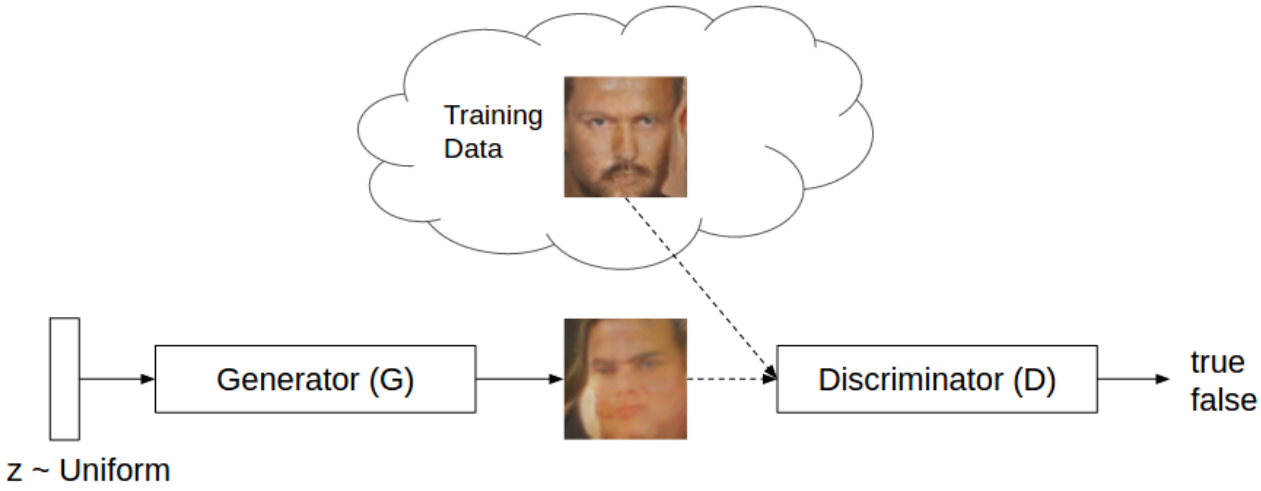
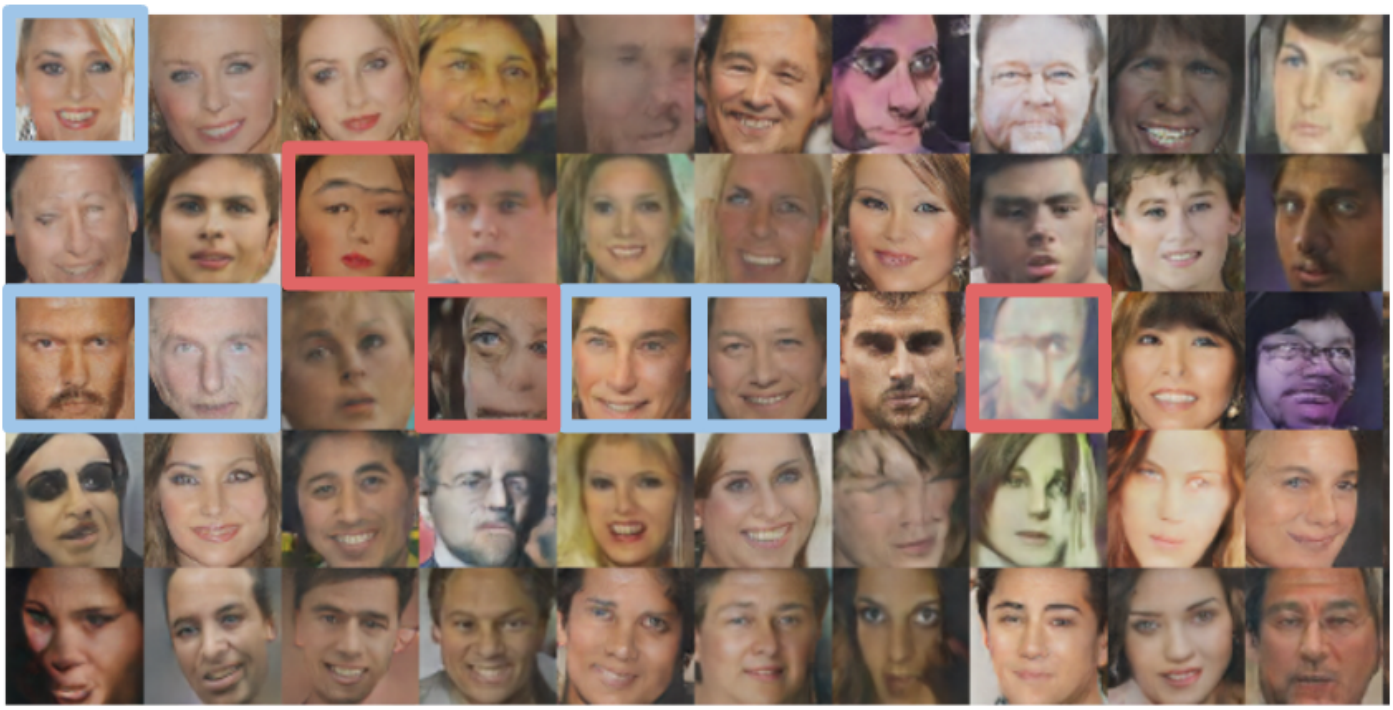


Image Generation: Adversarial Network (Goodfellow et al., Radford et al.)

(Radford et al, "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks", ICLR, 2016.)

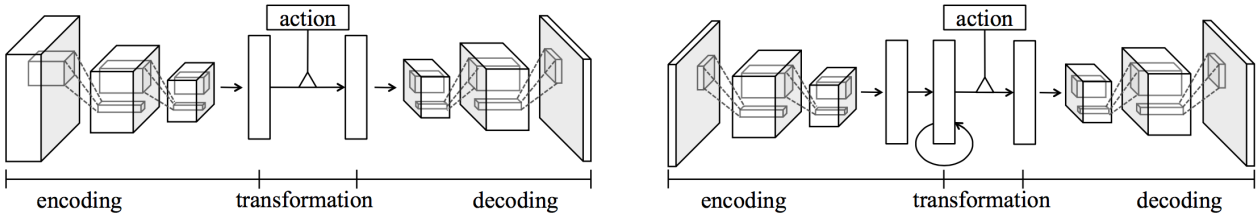
Image Generation: Adversarial Network (Goodfellow et al., Radford et al.)



(Radford et al, "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks", ICLR, 2016.)

Video Prediction (Oh et al.)

- Predict future frames given previous frames and actions in Atari games.



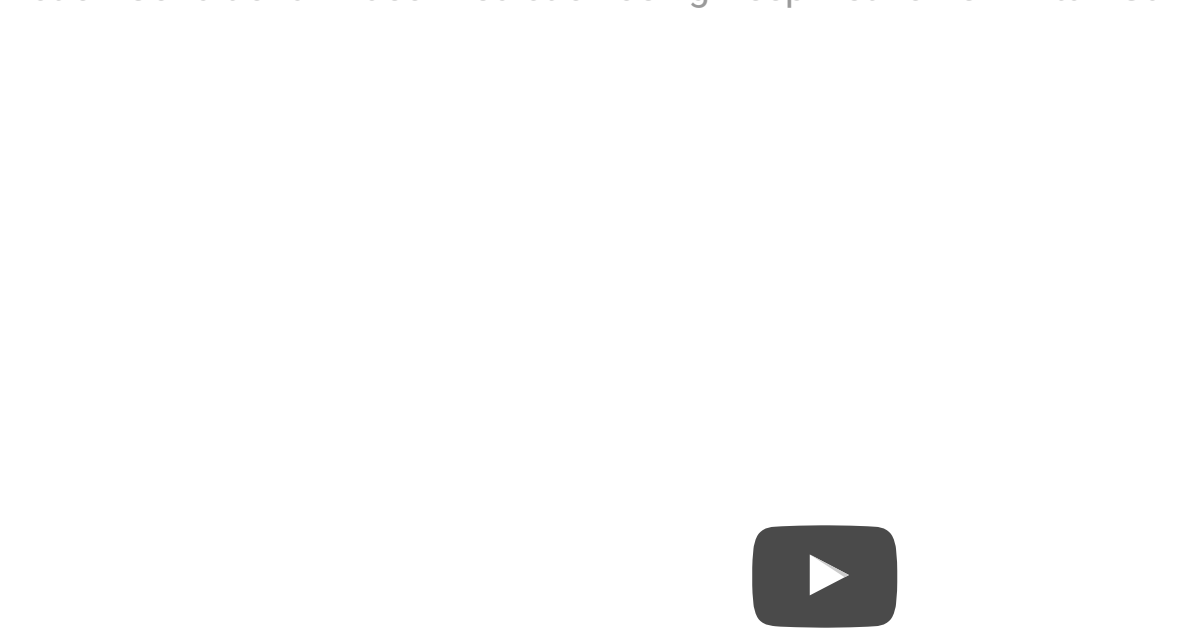
(a) Feedforward encoding (b) Recurrent encoding

(Oh et al, "Action-Conditional Video Prediction using Deep Networks in Atari Games", NIPS, 2015.)

Video Prediction (Oh et al.)

In [3]:
```
YouTubeVideo("7FBFhG2LgNQ", width=800, height=600)
```

Out[3]:



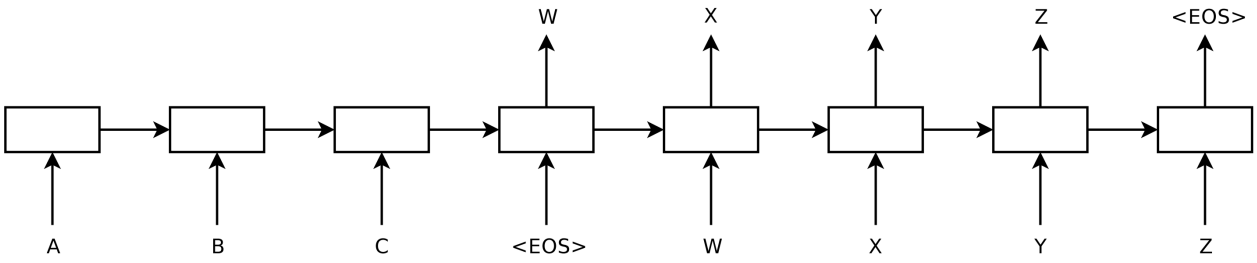
Outline

- Motivation
- Basics of Neural Networks
  - Forward Propagation
  - Backward Propagation
- Deep Neural Networks
  - Convolutional Neural Networks
  - Recurrent Neural Networks
- Applications
  - Computer Vision
  - **Natural Language Processing**
  - Reinforcement Learning

Sequence-to-Sequence Learning Framework (Sutskever et al.)

- A general RNN framework for sequence-to-sequence prediction

$$P\left(y_1, y_2, \ldots, y_{T'} \middle| x_1, x_2, \ldots, x_T\right)$$



(Sutskever et al, "Sequence to Sequence Learning with Neural Networks", NIPS, 2014.)

Seq2Seq: Application to Machine Translation (Sutskever et al.)

- Achieves the state-of-the-art results on English-to-French dataset.

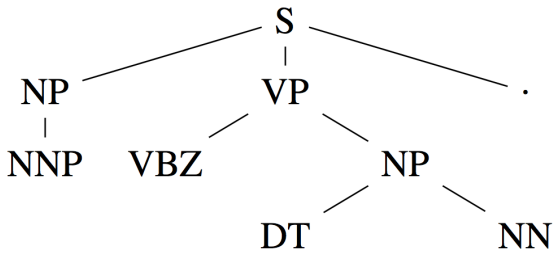
Method	test BLEU score (ntst14)
Baseline System [29]	33.30
Cho et al. [5]	34.54
Best WMT'14 result [9]	**37.0**
Rescoring the baseline 1000-best with a single forward LSTM	35.61
Rescoring the baseline 1000-best with a single reversed LSTM	35.85
Rescoring the baseline 1000-best with an ensemble of 5 reversed LSTMs	**36.5**
Oracle Rescoring of the Baseline 1000-best lists	∼45

(Sutskever et al, "Sequence to Sequence Learning with Neural Networks", NIPS, 2014.)

Seq2Seq: Application to Grammar Parsing (Vinyals et al.)

- A parsing tree can be represented as a sequence
- Seq2Seq framework can be applied (sentence → parse tree)

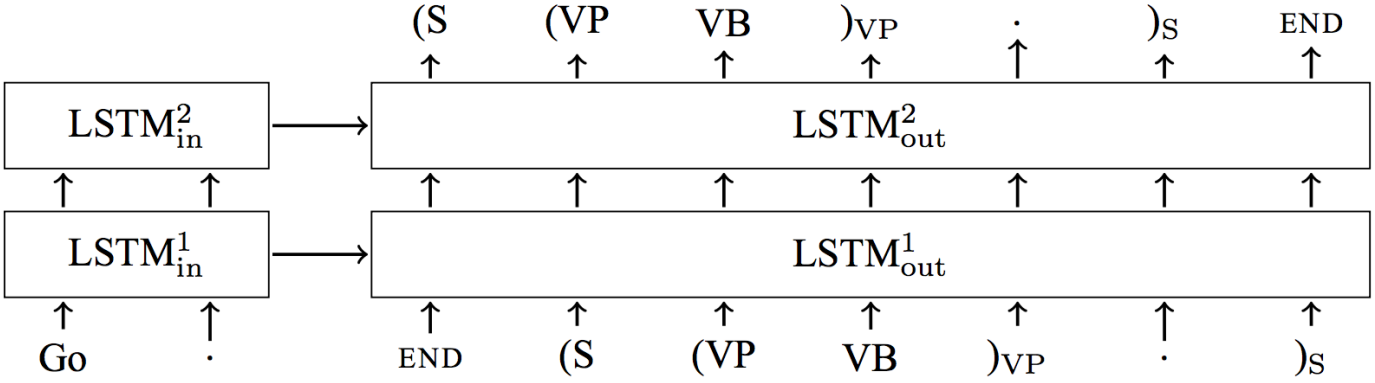
John has a dog . →



John has a dog . → (S (NP NNP)$_{\text{NP}}$ (VP VBZ (NP DT NN)$_{\text{NP}}$)$_{\text{VP}}$.)$_{\text{S}}$

(Vinyals et al, "Grammar as a Foreign Language", NIPS, 2015.)

Seq2Seq: Application to Grammar Parsing (Vinyals et al.)



(Vinyals et al, "Grammar as a Foreign Language", NIPS, 2015.)

Seq2Seq: Application to Program Execution (Zaremba et al.)

- Input: source code
- Output: execution result

Input:

```
print(398345+425098)
```

Target: 823443

(Zaremba et al., "Learning to Execute", ICLR, 2015.)

Seq2Seq: Application to Program Execution (Zaremba et al.)

Input:

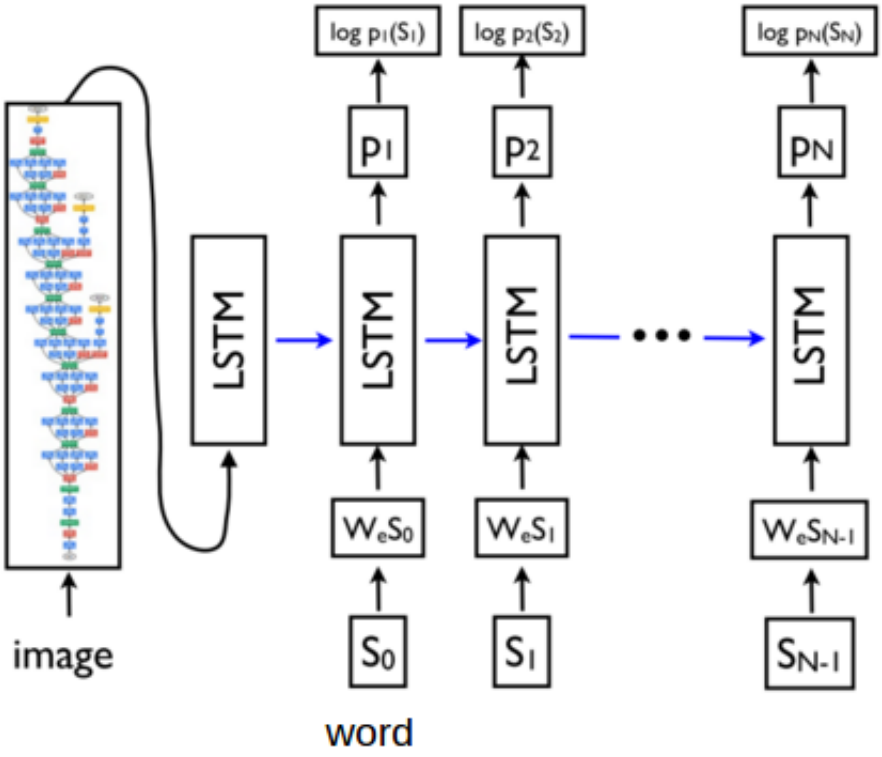
```
f=483654
for x in range(9):f-=913681
a=f
for x in range(12):a-=926785
print((124798 if a>326533 else 576599)).
```

Target:	576599.
"Baseline" prediction:	176599.
"Naive" prediction:	576599.
"Mix" prediction:	576599.
"Combined" prediction:	576599.

(Zaremba et al, "Learning to Execute", ICLR, 2015.)

Image Caption Generation (Vinyals et al.)

- Goal: generate a text that describes a given image
- Idea proposed by Vinyals et al.
  - Use a pre-trained CNN to extract image features
  - Condition a RNN for generating an image description.



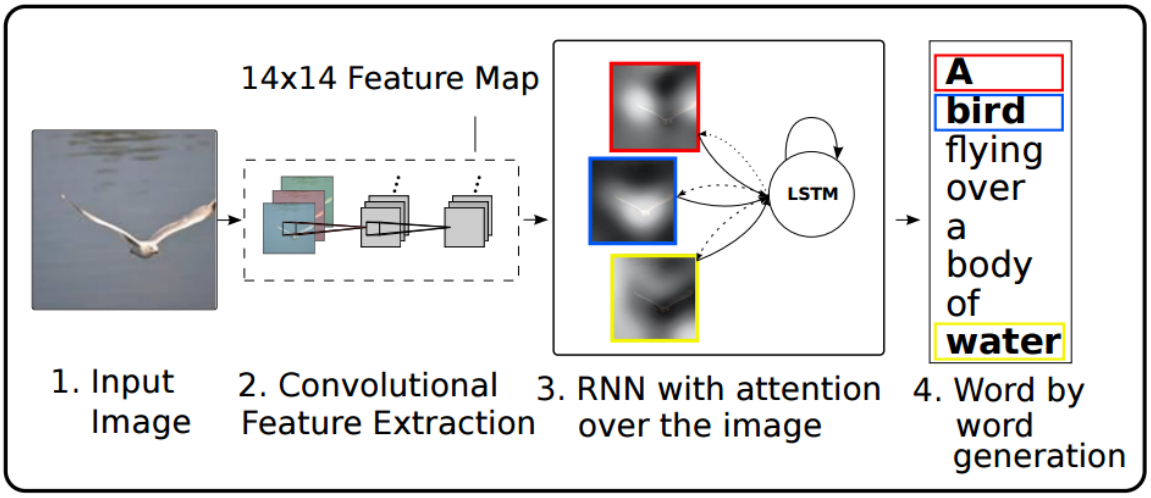
(Vinyals et al., "Show and Tell: A Neural Image Caption Generator", CVPR, 2015.)

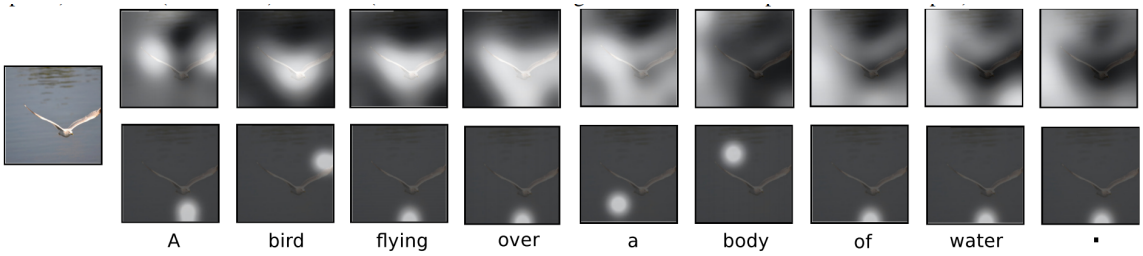
Image Caption Generation (Vinyals et al.)

(Vinyals et al, "Show and Tell: A Neural Image Caption Generator", CVPR, 2015.)

Image Caption Generation (Xu et al.)

- Another idea proposed by Xu et al.
  - Let the model pay attention to only a part of an image when generating a word.





(Xu et al, "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention", ICML, 2015.)

Image Caption Generation (Kiros et al.)

- Some examples from Kiros et al.

the two birds are trying to be seen in the water . (can't count)

a giraffe is standing next to a fence in a field . (hallucination)

a parked car while driving down the road . (contradiction)

the handlebars are trying to ride a bike rack . (nonsensical)

a woman and a bottle of wine in a garden . (gender)

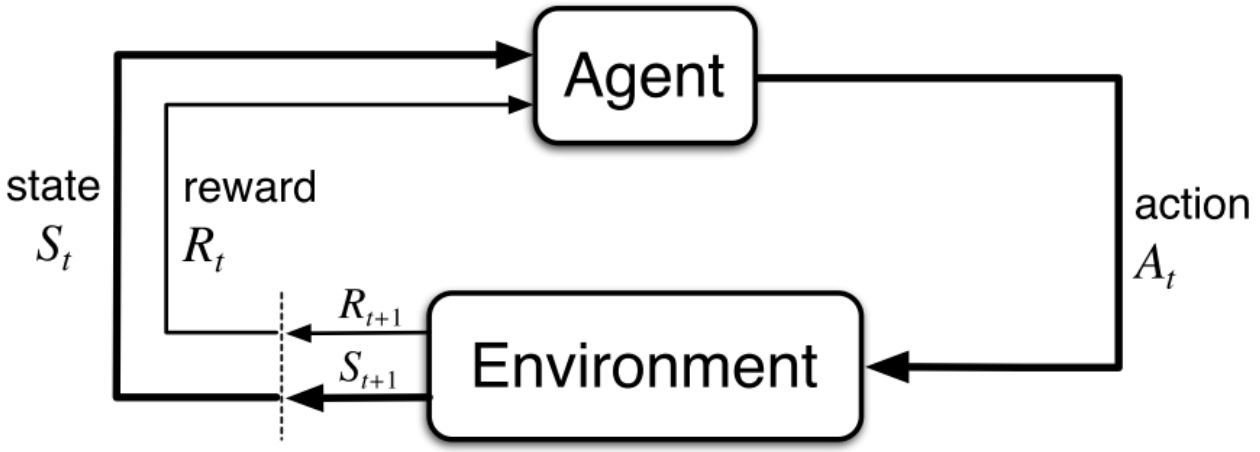
(Figures from Ruslan Salakhutdinov)

Outline

- Motivation
- Basics of Neural Networks
  - Forward Propagation
  - Backward Propagation
- Deep Neural Networks
  - Convolutional Neural Networks
  - Recurrent Neural Networks
- Applications
  - Computer Vision
  - Natural Language Processing
  - **Reinforcement Learning**

Reinforcement Learning

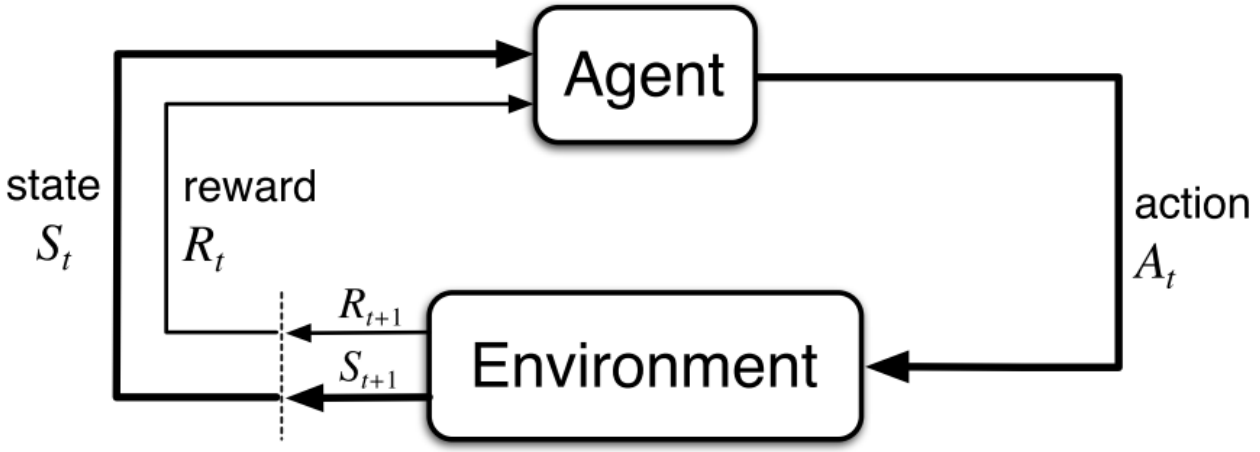
- An agent interacts with an environment to find an optimal policy.
- An agent observes a state s_t, chooses an action a_t, receives a reward r_t, and goes to the next state s_{t+1}.
- The goal is to maximize the total reward until the episode terminates (episodic task).



(Figure from "Reinforcement Learning: An Introduction" by Richard S. Sutton and Andrew G. Barto)

Brief Summary of Q-Learning

- $Q(s, a)$: expected future reward when choosing action a at state s.
- The agent learns to estimate $Q(s, a)$ by trial-and-error.
- Greedy policy: $\mathbf{argmax}_a Q(s, a)$
- Problem: need to approximate $Q(s, a)$ when the state space is very large.



(Figure from "Reinforcement Learning: An Introduction" by Richard S. Sutton and Andrew G. Barto)

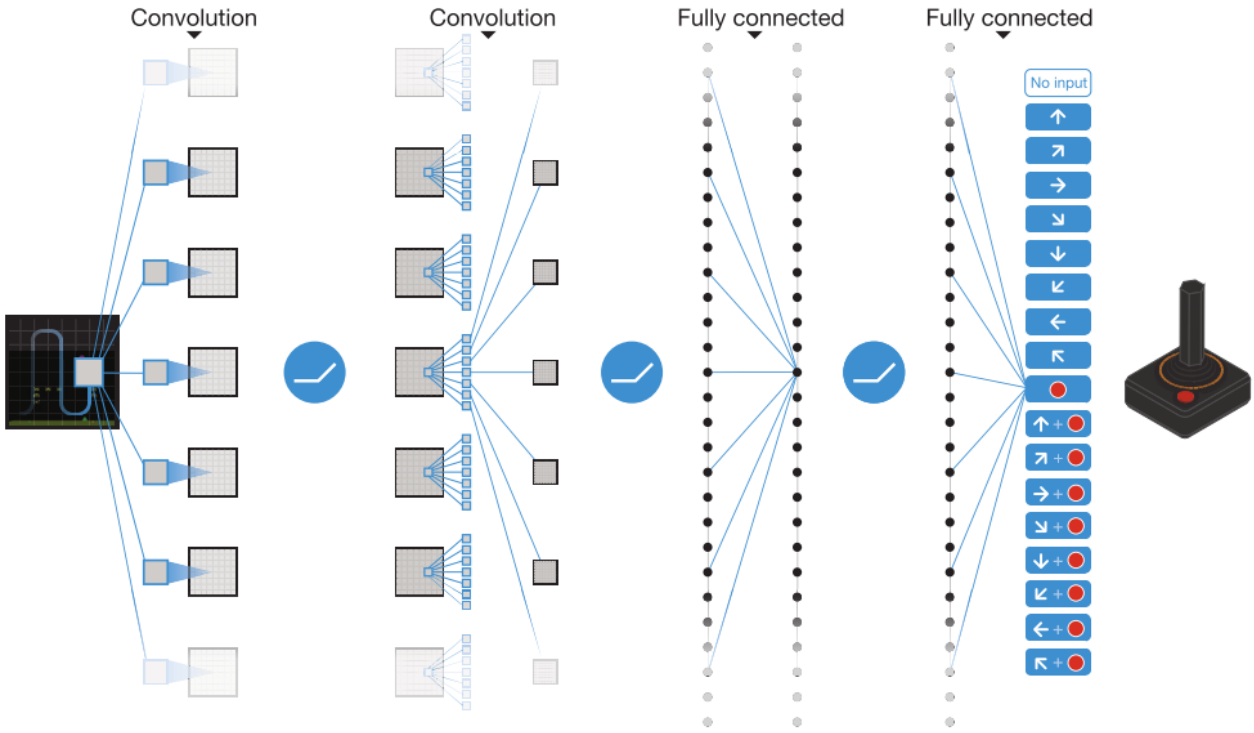
Deep Q-Network (Minh et al.)

- A recent breakthrough from Google DeepMind
- Achieves human-level performances on many Atari 2600 games

(Minh et al. "Human-level control through deep reinforcement learning", Nature, 2015.)

Deep Q-Network (Minh et al.)

- Key idea: Use a CNN to approximate Q-values



(Minh et al. "Human-level control through deep reinforcement learning", Nature, 2015.)

Deep Q-Network (Minh et al.)

In [4]:
```
YouTubeVideo("Q70ulPJW3Gk", width=800, height=600)
```

Out[4]:

DeepMind: Breakout

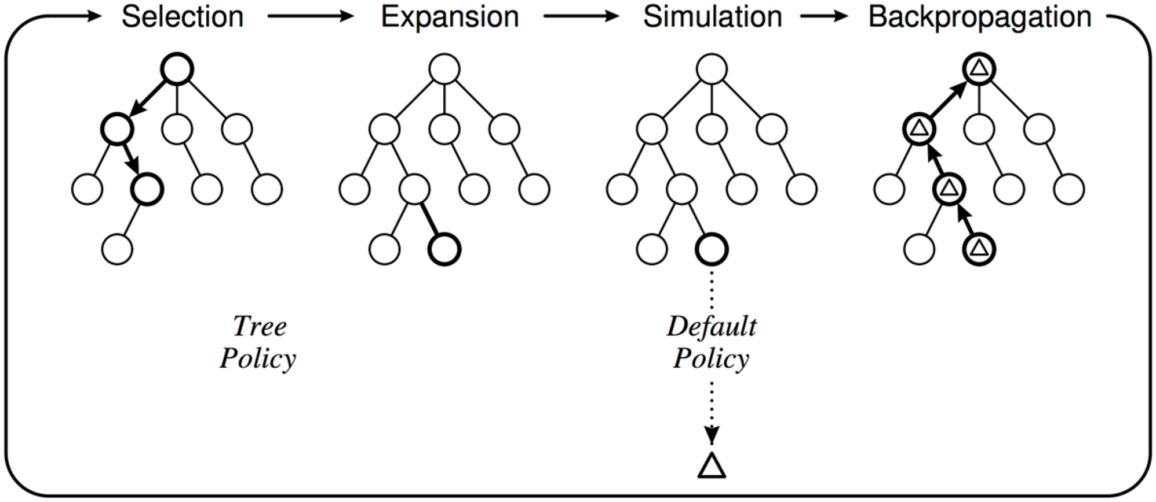
AlphaGo (Silver et al.)

- Another breakthrough from Google DeepMind
- Achieves super-human performance on Go

(Silver et al., "Mastering the game of Go with deep neural networks and tree search", Nature, 2016.)

Brief Summary of Monte-Carlo Tree Search (MTCS)

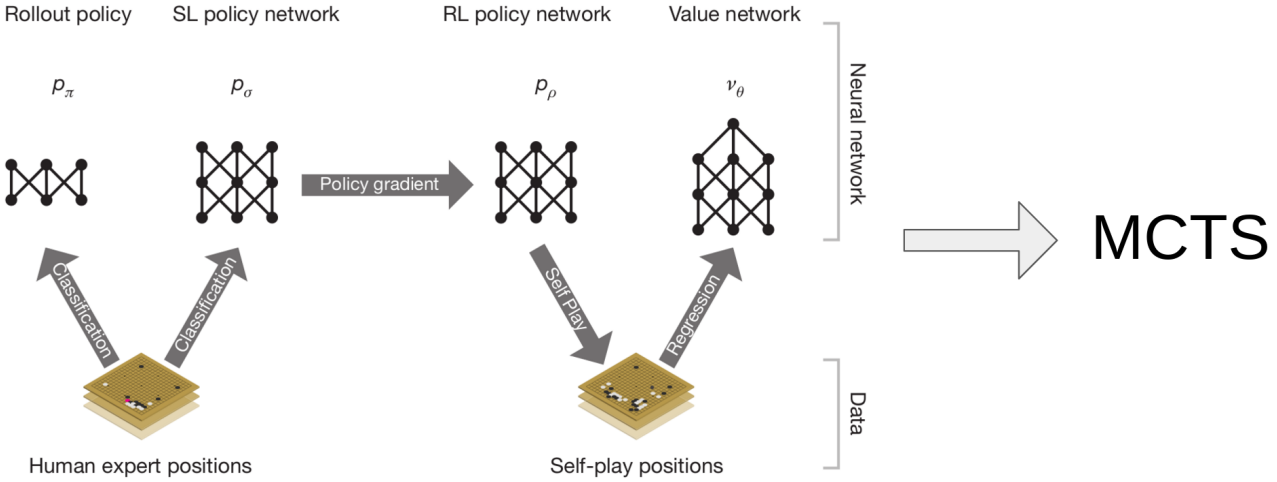
- Simulate many possible futures and choose the best action
- Problem: search space is too large!
  - Search over only a reasonable state space (tree policy should be reasonable).
  - Search up to a certain depth and use a default policy (usually random) to get the outcome.



(Figure from Browne et al.)

AlphaGo (Silver et al.)

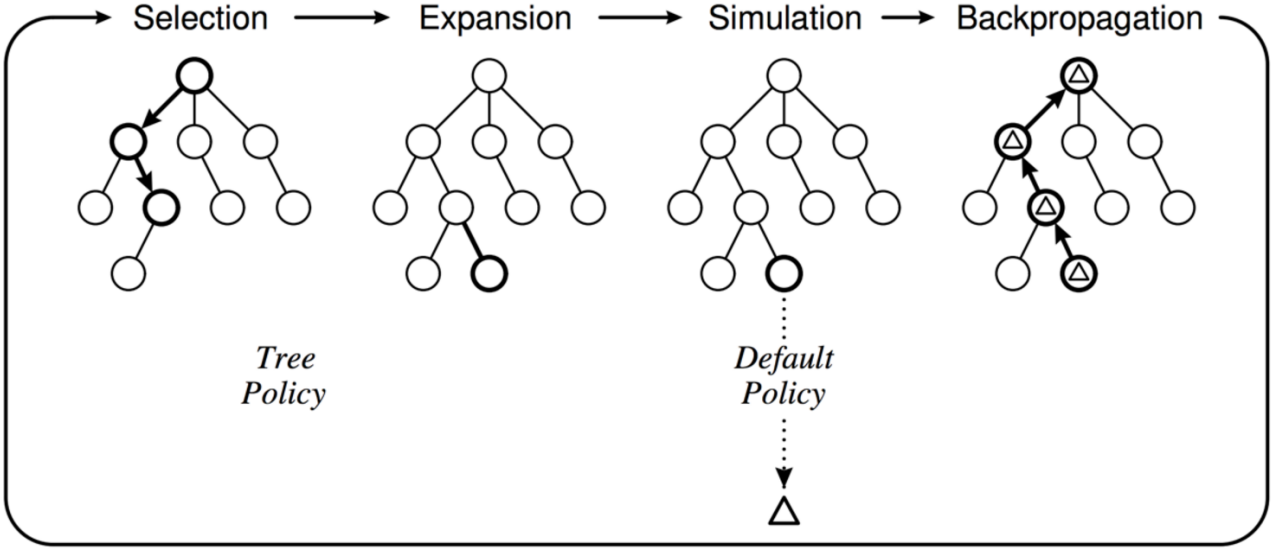
1. Supervised Learning: Train a **policy network** to predict human experts' moves.
2. Reinforcement Learning: Improve the policy network through self-play.
3. Reinforcement Learning: Train a **value network** to predict whether the agent wins at the end or not.
4. Use the learned networks to do MCTS more efficiently!



(Silver et al., "Mastering the game of Go with deep neural networks and tree search", Nature, 2016.)

AlphaGo (Silver et al.)

- Use the **policy network** as a prior distribution over actions.
  - Allows searching over only reasonable state spaces.
- Use the **value network** to directly predict the outcome at the leaf node
- Use a fshallow but fast policy network as a default policy (better than random)



(Figure from Browne et al.)

AlphaGo (Silver et al.)

- AlphaGo beat Lee Sedol (the world's top Go player).

Summary

- Deep Learning: machine learning algorithms based on learning multiple levels of representation.
- Neural networks can implement the idea of deep learning in a very flexible way.
- Deep neural networks (e.g., CNN, RNN) have made remarkable advances in computer vision, natural language processing, and reinforcement learning area.